Taming Past LTL and Flat Counter Systems

Stéphane Demri, Amit Kumar Dhar, Arnaud Sangnier

Taming Past LTL and Flat Counter Systems*

Stéphane Demri[1], Amit Kumar Dhar[2], and Arnaud Sangnier[2]

[1] LSV, CNRS, ENS Cachan, INRIA, France
[2] LIAFA, Univ Paris Diderot, Sorbonne Paris Cité, CNRS, France

Abstract. Reachability and LTL model-checking problems for flat counter systems are known to be decidable but whereas the reachability problem can be shown in NP, the best known complexity upper bound for the latter problem is made of a tower of several exponentials. Herein, we show that the problem is only NP-complete even if LTL admits past-time operators and arithmetical constraints on counters. Actually, the NP upper bound is shown by adequately combining a new stuttering theorem for Past LTL and the property of small integer solutions for quantifier-free Presburger formulae. Other complexity results are proved, for instance for restricted classes of flat counter systems.

1 Introduction

Flat counter systems. Counter systems are finite-state automata equipped with program variables (counters) interpreted over non-negative integers. They are used in many places like, broadcast protocols [9] and programs with pointers [12] to quote a few examples. But, alongwith their large scope of usability, many problems on general counter systems are known to be undecidable. Indeed, this computational model can simulate Turing machines. Decidability of reachability problems or model-checking problems based on temporal logics, can be regained by considering subclasses of counter systems, see e.g. [14]. An important and natural class of counter systems, in which various practical cases of infinite-state systems (e.g. broadcast protocols [11]) can be modelled, are those with a *flat* control graph, i.e, those where no control state occurs in more than one simple cycle, see e.g. [1,5,11,20]. Decidability results on verifying safety and reachability properties on flat counter systems have been obtained in [5,11,3]. However, so far, such properties have been rarely considered in the framework of any formal specification language (see an exception in [4]). In [7], a class of Presburger counter systems is identified for which the local model checking problem for Presburger-CTL* is shown decidable. These are Presburger counter systems defined over flat control graphs with arcs labelled by adequate Presburger formulae. Even though flatness is clearly a substantial restriction, it is shown in [20] that many classes of counter systems with computable Presburger-definable reachability sets are *flattable*, i.e. there exists a flat unfolding of the counter system with identical reachability sets. Hence, the possibility of flattening a counter system is strongly

* Supported by ANR project REACHARD ANR-11-BS02-001.

related to semilinearity of its reachability set. Moreover, in [4] model-checking relational counter systems over LTL formulae is shown decidable when restricted to flat formulae (their translation into automata leads to flat structures).
Towards the complexity of temporal model-checking flat counter systems. In [7], it is shown that CTL* model-checking over the class of so-called *admissible* counter systems is decidable by reduction into the satisfiability problem for Presburger arithmetic, the decidable first-order theory of natural numbers with addition. Obviously CTL* properties are more expressive than reachability properties but this has a cost. However, for the class of counter systems considered in this paper, this provides a very rough complexity upper bound in 4ExpTime. Herein, our goal is to revisit standard decidability results for subclasses of counter systems obtained by translation into Presburger arithmetic in order to obtain optimal complexity upper bounds.
Our contributions. In the paper, we establish several computational complexity characterizations of model-checking problems restricted to flat counter systems in the presence of a rich LTL-like specification language with arithmetical constraints and past-time operators. Not only we provide an optimal complexity but also, we believe that our proof technique could be reused for further extensions. Indeed, we combine three proof techniques: the general stuttering theorem [17], the property of small integer solutions of equation systems [2] (this latter technique is used since [24]) and the elimination of disjunctions in guards (see Section 5.2). Let us be a bit more precise.

We extend the stuttering principle established in [17] for LTL (without past-time operators) to Past LTL. The stuttering theorem from [17] for LTL without past-time operators has been used to show that LTL model-checking over *weak* Kripke structures is in NP [16] (weakness corresponds to flatness). It is worth noting that another way to show a similar result would be to eliminate past-time operators thanks to Gabbay's Separation Theorem [13] (preserving initial equivalence) but the temporal depth of formulae might increase at least exponentially, which is a crucial parameter in our complexity analysis. We show that the model-checking problem restricted to flat counter systems in the presence of LTL with past-time operators is in NP (Theorem 17) by combining the abovementioned proof techniques. Apart from the use of the general stuttering theorem (Theorem 3), we take advantage of the other properties stated for instance in Lemma 12 (characterization of runs by quantifier-free Presburger formulae) and Theorem 14 (elimination of disjunctions in guards preserving flatness). In the paper, complexity results for fragments/subproblems are also considered. For instance, we get a sharp lower bound since we establish that the model-checking problem on path schemas (a fundamental structure in flat counter systems) with only 2 loops is already NP-hard (see Lemma 11). A summary table can be found in Section 6.

Omitted proofs can be found in [6].

2 Flat Counter Systems and its LTL Dialect

We write $\mathbb{N}$ [resp. $\mathbb{Z}$] to denote the set of natural numbers [resp. integers] and $[i, j]$ to denote $\{k \in \mathbb{Z} : i \leq k \text{ and } k \leq j\}$. For $\boldsymbol{v} \in \mathbb{Z}^n$, $\boldsymbol{v}[i]$ denotes the i^{th} element of $\boldsymbol{v}$ for every $i \in [1, n]$. For some n-ary tuple t, we write $\pi_j(t)$ to denote the j^{th} element of t ($j \leq n$). In the sequel, integers are encoded with a binary representation. For a finite alphabet Σ, Σ^* represents the set of finite words over Σ, Σ^+ the set of finite non-empty words over Σ and Σ^ω the set of ω-words over Σ. For a finite word $w = a_1 \dots a_k$ over Σ, we write $\text{len}(w)$ to denote its *length* k. For $0 \leq i < \text{len}(w)$, $w(i)$ represents the $(i+1)$-th letter of the word, here a_{i+1}.

2.1 Counter Systems

Let $\mathtt{C} = \{\mathtt{x}_1, \mathtt{x}_2, \dots\}$ be a countably infinite set of *counters* (variables interpreted over non-negative integers) and $\text{AT} = \{p_1, p_2, \dots\}$ be a countable infinite set of propositional variables (abstract properties about program points). We write $\mathtt{C}_n$ to denote $\{\mathtt{x}_1, \mathtt{x}_2, \dots, \mathtt{x}_n\}$. The set $\mathtt{G}(\mathtt{C}_n)$ of *guards* (arithmetical constraints on counters in $\mathtt{C}_n$) is defined inductively as follows: $\mathtt{t} ::= a.\mathtt{x} \mid \mathtt{t} + \mathtt{t}$ and $\mathtt{g} ::= \mathtt{t} \sim b \mid \mathtt{g} \wedge \mathtt{g} \mid \mathtt{g} \vee \mathtt{g}$, where $\mathtt{x} \in \mathtt{C}_n$, $a \in \mathbb{Z}$, $b \in \mathbb{Z}$ and $\sim \in \{=, \leq, \geq, <, >\}$. Such guards are closed under negations (but negation is not part of the logical connectives) and the truth constants $\top$ and $\bot$ can be easily defined too. Given $\mathtt{g} \in \mathtt{G}(\mathtt{C}_n)$ and a vector $\mathbf{v} \in \mathbb{N}^n$, we say that $\mathbf{v}$ satisfies $\mathtt{g}$, written $\mathbf{v} \models \mathtt{g}$, if the formula obtained by replacing each $\mathtt{x}_i$ by $\boldsymbol{v}[i]$ holds.

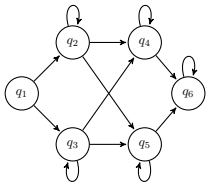
Definition 1 (Counter system). *For $n \geq 1$, a counter system S is a tuple $\langle Q, \mathtt{C}_n, \Delta, \mathbf{l}\rangle$ where Q is a finite set of* control states*, $\mathbf{l} : Q \to 2^{\text{AT}}$ is a* labelling function *and $\Delta \subseteq Q \times \mathtt{G}(\mathtt{C}_n) \times \mathbb{Z}^n \times Q$ is a finite set of edges labeled by guards and updates of the counter values (*transitions*).*

For $\delta = (q, \mathtt{g}, \mathbf{u}, q')$ in Δ, we use the following notations $\mathit{source}(\delta) = q$, $\mathit{target}(\delta) = q'$, $\mathit{guard}(\delta) = \mathtt{g}$ and $\mathit{update}(\delta) = \mathbf{u}$. As usual, to a counter system $S = \langle Q, \mathtt{C}_n, \Delta, \mathbf{l}\rangle$, we associate a labeled transition system $TS(S) = \langle C, \to\rangle$ where $C = Q \times \mathbb{N}^n$ is the set of *configurations* and $\to \subseteq C \times \Delta \times C$ is the *transition relation* defined by: $\langle\langle q, \mathbf{v}\rangle, \delta, \langle q', \mathbf{v}'\rangle\rangle \in \to$ (also written $(q, \mathbf{v}) \xrightarrow{\delta} (q', \mathbf{v}')$) iff $q = \mathit{source}(\delta)$, $q' = \mathit{target}(\delta)$, $\mathbf{v} \models \mathit{guard}(\delta)$ and $\mathbf{v}' = \mathbf{v} + \mathit{update}(\delta)$. In such a transition system, the counter values are non-negative since $C = Q \times \mathbb{N}^n$. We extend the transition relation $\to$ to finite words of transitions in Δ^+ as follows. For each $w = \delta_1\delta_2 \dots \delta_\alpha \in \Delta^+$, we have $\langle q, \mathbf{v}\rangle \xrightarrow{w} \langle q', \mathbf{v}'\rangle$ if there are $c_0, c_1, \dots, c_{\alpha+1} \in C$ such that $c_i \xrightarrow{\delta_i} c_{i+1}$ for all $i \in [0, \alpha]$, $c_0 = (q, \mathbf{v})$ and $c_{\alpha+1} = \langle q', \mathbf{v}'\rangle$. We say that an ω-word $w \in \Delta^\omega$ is *fireable* in S from a configuration $c_0 \in Q \times \mathbb{N}^n$ if for all finite prefixes w' of w there exists a configuration $c \in Q \times \mathbb{N}^n$ such that $c_0 \xrightarrow{w'} c$. We write $\mathit{lab}(c_0)$ to denote the set of ω-words (*labels*) which are fireable from c_0 in S.

Given a configuration $c_0 \in Q \times \mathbb{N}^n$, a *run* ρ starting from c_0 in S is an infinite path in the associated transition system $TS(S)$ denoted as: $\rho := c_0 \xrightarrow{\delta_0}$

$\cdots \xrightarrow{\delta_{\alpha-1}} c_\alpha \xrightarrow{\delta_\alpha} \cdots$ where $c_i \in Q \times \mathbb{N}^n$ and $\delta_i \in \Delta$ for all $i \in \mathbb{N}$. Let $lab(\rho)$ be the ω-word $\delta_0\delta_1\ldots$ associated to the run ρ. Note that by definition we have $lab(\rho) \in lab(c_0)$. When E is an ω-regular expression over the finite alphabet Δ and c_0 is an initial configuration, $lab(E, c_0)$ is defined as the set of labels of infinite runs ρ starting at c_0 such that $lab(\rho)$ belongs to the language defined by E. So $lab(E, c_0) \subseteq lab(c_0)$.

We say that a counter system is *flat* if every node in the underlying graph belongs to at most one simple cycle (a cycle being simple if no edge is repeated twice in it) [5]. In a flat counter system, simple cycles can be organized as a DAG where two simple cycles are in the relation whenever there is path between a node of the first cycle and a node of the second cycle. We denote by $\mathcal{CFS}$ the class of flat counter systems.



On the left, we present the control graph of a flat counter system (guards and updates are omitted). A *Kripke structure* S is a tuple $\langle Q, \Delta, \mathbf{l}\rangle$ where $\Delta \subseteq Q \times Q$ and $\mathbf{l}$ is labelling. It can be viewed as a degenerate form of counter systems without counters (in the sequel, we take the freedom to see them as counter systems). All standard notions on counter systems naturally apply to Kripke structures too (configuration, run, flatness, etc.). In the sequel, we shall also investigate the complexity of model-checking problems on flat Kripke structures (such a class is denoted by $\mathcal{KFS}$).

2.2 Linear Temporal Logic with Past and Arithmetical Constraints

Model-checking problem for Past LTL over finite state systems is known to be PSPACE-complete. In spite of this nice feature, a propositional variable p only represents an abstract property about the current configuration of the system. A more satisfactory solution is to include in the logical language the possibility to express directly constraints between variables of the program, whence giving up the standard abstraction made with propositional variables. We define below a version of LTL dedicated to counter systems in which the atomic formulae are linear constraints; this is analogous to the use of concrete domains in description logics [21]. Note that capacity constraints from [8] are arithmetical constraints different from those defined below. Formulae of PLTL[$\mathtt{C}$] are defined from $\phi ::= p \mid \mathtt{g} \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \mathtt{X}\phi \mid \phi\mathtt{U}\phi \mid \mathtt{X}^{-1}\phi \mid \phi\mathtt{S}\phi$ where $p \in$ AT and $\mathtt{g} \in \mathtt{G}(\mathtt{C}_n)$ for some n. We may use the standard abbreviations $\mathtt{F}$, $\mathtt{G}$, $\mathtt{G}^{-1}$ etc. For instance, the formula $\mathtt{GF}(\mathtt{x}_1 + 2 \geq \mathtt{x}_2)$ states that infinitely often the value of counter 1 plus 2 is greater than the value of counter 2. The past-time operators $\mathtt{S}$ and $\mathtt{X}^{-1}$ do not add expressive power to the logic itself, but it is known that it helps a lot to express properties succinctly, see e.g. [19,18]. The temporal depth of ϕ, written $td(\phi)$, is defined as the maximal number of imbrications of temporal operators in ϕ. Restriction of PLTL[$\mathtt{C}$] to atomic formulae from AT only is written PLTL[$\emptyset$], standard version of LTL with past-time operators. Models of PLTL[$\mathtt{C}$] are essentially abstractions of runs from counter systems,

i.e. ω-sequences $\sigma : \mathbb{N} \to 2^{\mathrm{AT}} \times \mathbb{N}^{\mathtt{C}}$. Given a model σ and a position $i \in \mathbb{N}$, the satisfaction relation $\models$ for PLTL[$\mathtt{C}$] is defined as follows (other cases can be defined similarly, see e.g. [18]):

- $\sigma, i \models\ p \overset{\text{def}}{\Leftrightarrow} p \in \pi_1(\sigma(i))$, $\sigma, i \models\ \mathtt{g} \overset{\text{def}}{\Leftrightarrow} \mathbf{v}_i \models \mathtt{g}$ where $\mathbf{v}_i[j] \overset{\text{def}}{=} \pi_2(\sigma(i))(\mathtt{x}_j)$,
- $\sigma, i \models\ \mathtt{X}\phi \overset{\text{def}}{\Leftrightarrow} \sigma, i+1 \models\ \phi$,
- $\sigma, i \models\ \phi_1 \mathtt{S} \phi_2 \overset{\text{def}}{\Leftrightarrow} \sigma, j \models\ \phi_2$ for some $0 \leq j \leq i$ s.t. $\sigma, k \models\ \phi_1, \forall j < k \leq i$.

Given $\langle Q, \mathtt{C}_n, \Delta, \mathbf{l} \rangle$ and a run $\rho := \langle q_0, \mathbf{v}_0 \rangle \xrightarrow{\delta_0} \cdots \xrightarrow{\delta_{p-1}} \langle q_p, \mathbf{v}_p \rangle \xrightarrow{\delta_p} \cdots$, we consider the model $\sigma_\rho : \mathbb{N} \to 2^{\mathrm{AT}} \times \mathbb{N}^{\mathtt{C}}$ such that $\pi_1(\sigma_\rho(i)) \overset{\text{def}}{=} \mathbf{l}(q_i)$ and $\pi_2(\sigma_\rho(i))(\mathtt{x}_j) \overset{\text{def}}{=} \mathbf{v}_i[j]$ for all $j \in [1, n]$ and all $i \in \mathbb{N}$. Note that $\pi_2(\sigma_\rho(i))(\mathtt{x}_j)$ is arbitrary for $j \notin [1, n]$. As expected, we extend the satisfaction relation to runs so that $\rho, i \models \phi \overset{\text{def}}{\Leftrightarrow} \sigma_\rho, i \models \phi$ whenever ϕ is built from counters in $\mathtt{C}_n$.

Given a fragment L of PLTL[$\mathtt{C}$] and a class $\mathcal{C}$ of counter systems, we write MC(L, $\mathcal{C}$) to denote the existential model checking problem: given $S \in \mathcal{C}$, a configuration c_0 and $\phi \in$ L, does there exist ρ starting from c_0 such that $\rho, 0 \models \phi$? In that case, we write $S, c_0 \models \phi$. It is known that for the full class of counter systems, the model-checking problem is undecidable, see e.g. [22]. Some restrictions, such as flatness, can lead to decidability as shown in [7] but the decision procedure there involves an exponential reduction to Presburger Arithmetic, whence the high complexity.

Theorem 2. *[7,16]* MC(PLTL[$\mathtt{C}$], $\mathcal{CFS}$) *can be solved in* 4ExpTime. MC(PLTL[$\emptyset$], $\mathcal{KFS}$) *restricted to formulae with temporal operators* U,X *is* NP*-complete.*

Our main goal is to characterize the complexity of MC(PLTL[$\mathtt{C}$], $\mathcal{CFS}$).

3 Stuttering Theorem for PLTL[$\emptyset$]

Stuttering of finite words or single letters has been instrumental to show results about the expressive power of PLTL[$\emptyset$] fragments, see e.g. [23,17]; for instance, PLTL[$\emptyset$] restricted to the temporal operator U characterizes the class of formulae defining classes of models invariant under stuttering. This is refined in [17] for PLTL[$\emptyset$] restricted to U and X, by taking into account not only the U-depth but also the X-depth of formulae and by introducing a principle of stuttering that involves both letter stuttering and word stuttering. In this section, we establish another substantial generalization that involves PLTL[$\emptyset$] with past-time temporal operators. Roughly speaking, we show that if $\sigma_1 \mathbf{s}^M \sigma_2, 0 \models \phi$ where $\sigma_1 \mathbf{s}^M \sigma_2$ is a PLTL[$\emptyset$] model ($\sigma_1, \mathbf{s}$ being finite words), $\phi \in$ PLTL[$\emptyset$], $td(\phi) \leq N$ and $M \geq 2N+1$, then $\sigma_1 \mathbf{s}^{2N+1} \sigma_2, 0 \models \phi$ (and other related properties). This extends a result without past-time operators [16]. Moreover, this turns out to be a key property (Theorem 3) to establish the NP upper bound even in the presence of counters. Note that Theorem 3 below is interesting for its own sake, independently of our investigation on flat counter systems. By lack of space, we state below the main definitions and result.

Given $M, M', N \in \mathbb{N}$, we write $M \approx_N M'$ iff $\texttt{Min}(M, N) = \texttt{Min}(M', N)$. Given $w = w_1 u^M w_2, w' = w_1 u^{M'} w_2 \in \Sigma^\omega$ and $i, i' \in \mathbb{N}$, we define an equivalence relation $\langle w, i\rangle \approx_N \langle w', i'\rangle$ (implicitly parameterized by w_1, w_2 and u) such that $\langle w, i\rangle \approx_N \langle w', i'\rangle$ means that the number of copies of u before position i and the number of copies of u before position i' are related by $\approx_N$ and the same applies for the number of copies after the positions. Moreover, if i and i' occur in the part where u is repeated, then they correspond to identical positions in u. More formally, $\langle w, i\rangle \approx_N \langle w', i'\rangle \stackrel{\text{def}}{\Leftrightarrow} M \approx_{2N} M'$ and one of the conditions holds true: (1) $i, i' < \text{len}(w_1) + N \cdot \text{len}(u)$ and $i = i'$, (2) $i \geq \text{len}(w_1) + (M - N) \cdot \text{len}(u)$, $i' \geq \text{len}(w_1) + (M' - N) \cdot \text{len}(u)$ and $(i - i') = (M - M') \cdot \text{len}(u)$, (3) $\text{len}(w_1) + N \cdot \text{len}(u) \leq i < \text{len}(w_1) + (M - N) \cdot \text{len}(u)$, $\text{len}(w_1) + N \cdot \text{len}(u) \leq i' < \text{len}(w_1) + (M' - N) \cdot \text{len}(u)$ and $|i - i'| = 0 \mod \text{len}(u)$. We state our stuttering theorem for PLTL[$\emptyset$] that is tailored for our future needs.

Theorem 3 (Stuttering). *Let $\sigma = \sigma_1 \mathbf{s}^M \sigma_2, \sigma' = \sigma_1 \mathbf{s}^{M'} \sigma_2 \in (2^{\text{AT}})^\omega$ and $i, i' \in \mathbb{N}$ such that $N \geq 2$, $M, M' \geq 2N + 1$ and $\langle \sigma, i\rangle \approx_N \langle \sigma', i'\rangle$. Then, for every* PLTL[$\emptyset$] *formula ϕ with $td(\phi) \leq N$, we have $\sigma, i \models \phi$ iff $\sigma', i \models \phi$.*

Proof. (sketch) The proof is by structural induction on the formula but first we need to establish properties whose proofs can be found in [6]. Let $w = w_1 u^M w_2, w' = w_1 u^{M'} w_2 \in \Sigma^\omega$, $i, i' \in \mathbb{N}$ and $N \geq 2$ such that $M, M' \geq 2N + 1$ and $\langle w, i\rangle \approx_N \langle w', i'\rangle$. We can show the following properties:

(Claim 1) $\langle w, i\rangle \approx_{N-1} \langle w', i'\rangle$ and $w(i) = w'(i')$.
(Claim 2) $\langle w, i + 1\rangle \approx_{N-1} \langle w', i' + 1\rangle$ and $i, i' > 0$ implies $\langle w, i - 1\rangle \approx_{N-1} \langle w', i' - 1\rangle$.
(Claim 3) For all $j \geq i$, there is $j' \geq i'$ such that $\langle w, j\rangle \approx_{N-1} \langle w', j'\rangle$ and for all $k' \in [i', j' - 1]$, there is $k \in [i, j - 1]$ such that $\langle w, k\rangle \approx_{N-1} \langle w', k'\rangle$.
(Claim 4) For all $j \leq i$, there is $j' \leq i'$ such that $\langle w, j\rangle \approx_{N-1} \langle w', j'\rangle$ and for all $k' \in [j' - 1, i']$, there is $k \in [j - 1, i]$ such that $\langle w, k\rangle \approx_{N-1} \langle w', k'\rangle$.

By way of example, let us present the induction step for subformulae of the form $\psi_1 \mathtt{U} \psi_2$. We show that $\sigma, i \models \psi_1 \mathtt{U} \psi_2$ implies $\sigma', i' \models \psi_1 \mathtt{U} \psi_2$. Suppose there is $j \geq i$ such that $\sigma, j \models \psi_2$ and for every $k \in [i, j - 1]$, we have $\sigma, k \models \psi_1$. There is $j' \geq i'$ satisfying (Claim 3). Since $td(\psi_1), td(\psi_2) \leq N - 1$, by (IH), we have $\sigma', j' \models \psi_2$. Moreover, for every $k' \in [i', j' - 1]$, there is $k \in [i, j - 1]$ such that $\langle w, k\rangle \approx_{N-1} \langle w', k'\rangle$ and by (IH), we have $\sigma', k' \models \psi_1$ for every $k' \in [i', j' - 1]$. Hence, $\sigma', i' \models \psi_1 \mathtt{U} \psi_2$. □

An alternative proof consists in using Ehrenfeucht-Fraïssé games [10].

4 Fundamental Structures: Minimal Path Schemas

In this section, we introduce the notion of a fundamental structure for flat counter systems, namely a path schema. Indeed, every flat counter system can be decomposed into a finite set of minimal path schemas and there are only an exponential number of them. So, all our nondeterministic algorithms on flat counter systems have a preliminary step that first guesses a minimal path schema.

4.1 Minimal Path Schemas

Let $S = \langle Q, \mathtt{C}_n, \Delta, \mathbf{l} \rangle$ be a flat counter system. A *path segment* p of S is a finite sequence of transitions from Δ such that $\mathit{target}(p(i)) = \mathit{source}(p(i+1))$ for all $0 \leq i < \mathrm{len}(p) - 1$. We write $\mathit{first}(p)$ [resp. $\mathit{last}(p)$] to denote the first [resp. last] control state of a path segment, in other words $\mathit{first}(p) = \mathit{source}(p(0))$ and $\mathit{last}(p) = \mathit{target}(p(\mathrm{len}(p) - 1))$. We also write $\mathit{effect}(p)$ to denote the sum vector $\sum_{0 \leq i < \mathrm{len}(p)} \mathit{update}(p(i))$ representing the total effect of the updates along the path segment. A path segment p is said to be *simple* if $\mathrm{len}(p) > 0$ and for all $0 \leq i, j < \mathrm{len}(p)$, $p(i) = p(j)$ implies $i = j$ (no repetition of transitions). A *loop* is a simple path segment p such that $\mathit{first}(p) = \mathit{last}(p)$. A *path schema* P is an ω-regular expression built over Δ such that its language represents an overapproximation of the set of labels obtained from infinite runs following the transitions of P. A path schema P is of the form $p_1 l_1^+ p_2 l_2^+ \dots p_k l_k^\omega$ where (1) l_1, $\dots$, l_k are loops and (2) $p_1 l_1 p_2 l_2 \dots p_k l_k$ is a path segment.

We write $\mathrm{len}(P)$ to denote $\mathrm{len}(p_1 l_1 p_2 l_2 \dots p_k l_k)$ and $\mathrm{nbloops}(P)$ as its number k of loops. Let $\mathcal{L}(P)$ denote the set of infinite words in Δ^ω which belong to the language defined by P. Note that some elements of $\mathcal{L}(P)$ may not correspond to any run because of constraints on counter values. Given $w \in \mathcal{L}(P)$, we write $\mathit{iter}_P(w)$ to denote the unique tuple in $(\mathbb{N} \setminus \{0\})^{k-1}$ such that $w = p_1 l_1^{\mathit{iter}_P(w)[1]} p_2 l_2^{\mathit{iter}_P(w)[2]} \dots p_k l_k^\omega$. So, for every $i \in [1, k-1]$, $\mathit{iter}_P(w)[i]$ is the number of times the loop l_i is taken. Then, for a configuration c_0, the set $\mathit{iter}_P(c_0)$ is the set of vectors $\{\mathit{iter}_P(w) \in (\mathbb{N} \setminus \{0\})^{k-1} \mid w \in \mathit{lab}(P, c_0)\}$. Finally, we say that a run ρ starting in a configuration c_0 *respects* a path schema P if $\mathit{lab}(\rho) \in \mathit{lab}(P, c_0)$ and for such a run, we write $\mathit{iter}_P(\rho)$ to denote $\mathit{iter}_P(\mathit{lab}(\rho))$. Note that by definition, if ρ respects P, then each loop l_i is visited at least once, and the last one infinitely.

So far, a flat counter system may have an infinite set of path schemas. However, we can impose minimality conditions on path schemas without sacrifying completeness. A path schema $p_1 l_1^+ p_2 l_2^+ \dots p_k l_k^\omega$ is *minimal* whenever $p_1 \cdots p_k$ is either the empty word or a simple non-loop segment, and l_1, $\dots$, l_k are loops with disjoint sets of transitions.

Lemma 4. *Given a flat counter system $S = \langle Q, \mathtt{C}_n, \Delta, \mathbf{l} \rangle$, the total number of minimal path schemas of S is finite and is smaller than* $\mathrm{card}(\Delta)^{(2 \times \mathrm{card}(\Delta))}$.

This is a simple consequence of the fact that in a minimal path schema, each transition occurs at most twice. In Figure 1, we present a flat counter system S with a unique counter and one of its minimal path schemas. Each transition δ_i labelled by $+i$ corresponds to a transition with the guard $\top$ and the update value $+i$. The minimal path schema shown in Figure 1 corresponds to the ω-regular expression $\delta_1(\delta_2\delta_3)^+\delta_4\delta_5(\delta_6\delta_5)^\omega$. Note that in the representation of path schemas, a state may occur several times, as it is the case for q_3 (this cannot occur in the representation of counter systems). Minimal path schemas play a crucial role in the sequel. Indeed, given a path schema P, there is a minimal path schema P' such that every run respecting P respects P' too. This can be easily

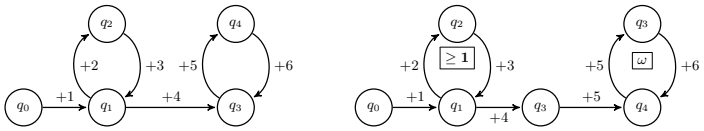


Fig. 1. A flat counter system and one of its minimal path schemas

shown since whenever a maximal number of copies of a simple loop is identified as a factor of $p_1 l_1 \cdots p_k l_k$, this factor is replaced by the simple loop unless it is already present in the path schema.

Finally, the conditions imposed on the structure of path schemas implies the following corollary which states that the number of minimal path schemas for a given flat counter system is at most exponential in the size of the system (see similar statements in [20]).

Corollary 5. *Given a flat counter system S and a configuration c_0, there is a finite set of minimal path schemas X of cardinality at most* $\mathrm{card}(\Delta)^{(2\times \mathrm{card}(\Delta))}$ *such that* $lab(c_0) = lab(\bigcup_{P\in X} P, c_0)$.

4.2 Complexity Results

We write $\mathcal{CPS}$ [resp. $\mathcal{KPS}$] to denote the class of path schemas from counter systems [resp. the class of path schemas from Kripke structures]. As a preliminary step, we consider the problem $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KPS})$ that takes as inputs a path schema P in $\mathcal{KPS}$, and $\phi \in \mathrm{PLTL}[\emptyset]$ and asks whether there is a run respecting P that satisfies ϕ. Let ρ and ρ' be runs respecting P. For $\alpha \geq 0$, we write $\rho \equiv_\alpha \rho' \stackrel{\text{def}}{\Leftrightarrow}$ for every $i \in [1, \mathrm{nbloops}(P) - 1]$, we have $\mathtt{Min}(iter_P(\rho)[i], \alpha) = \mathtt{Min}(iter_P(\rho')[i], \alpha)$. We state below a result concerning the runs of flat counter systems when respecting the same path schema.

Proposition 6. *Let S be a flat counter system, P be a path schema, and $\phi \in \mathrm{PLTL}[\emptyset]$. For all runs ρ and ρ' respecting P such that $\rho \equiv_{2td(\phi)+5} \rho'$, we have $\rho, 0 \models \phi$ iff $\rho', 0 \models \phi$.*

This property can be proved by applying Theorem 3 repeatedly in order to get rid of the unwanted iterations of the loops. Our algorithm for $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KPS})$ takes advantage of a result from [18] for model-checking ultimately periodic models with formulae from Past LTL. An *ultimately periodic path* is an infinite word in Δ^ω of the form uv^ω were uv is a path segment and consequently $first(v) = last(v)$. According to [18], given an ultimately periodic path w, and a formula $\phi \in \mathrm{PLTL}[\emptyset]$, the problem of checking whether there exists a run ρ such that $lab(\rho) = w$ and $\rho, 0 \models \phi$ is in PTime (a tighter bound of NC can be obtained by combining results from [15] and Theorem 3).

Lemma 7. $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KPS})$ *is in* NP.

The proof is a consequence of Proposition 6 and [18]. Indeed, given $\phi \in \mathrm{PLTL}[\emptyset]$ and $P = p_1 l_1^+ p_2 l_2^+ \dots p_k l_k^\omega$, first guess $\mathbf{m} \in [1, 2td(\phi)+5]^{k-1}$ and check whether $\rho, 0 \models \phi$ where ρ is the obvious ultimately periodic word such that $lab(\rho) = p_1 l_1^{\mathbf{m}[1]} p_2 l_2^{\mathbf{m}[2]} \dots p_k l_k^\omega$. Since $\mathbf{m}$ is of polynomial size and $\rho, 0 \models \phi$ can be checked in polynomial time by [18], we get the NP upper bound.

From [16], we have the lower bound for $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KPS})$.

Lemma 8. *[16]* $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KPS})$ *is* NP*-hard even if restricted to* `X` *and* `F`*.*

For a fixed $n \in \mathbb{N}$, we write $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KPS}(n))$ to denote the restriction of $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KPS})$ to path schemas with at most n loops. When n is fixed, the number of ultimately periodic paths w in $\mathcal{L}(P)$ such that each loop (except the last one) is visited is at most $2td(\phi)+5$ times is bounded by $(2td(\phi)+5)^n$, which is polynomial in the size of the input (because n is fixed).

Theorem 9. $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KPS})$ *is* NP*-complete.*
Given a fixed $n \in \mathbb{N}$, $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KPS}(n))$ *is in* PTime.

Note that it can be proved that $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KPS}(n))$ is in NC, hence giving a tighter upper bound for the problem. This can be obtained by observing that we can run the NC algorithm for model checking $\mathrm{PLTL}[\emptyset]$ over ultimately periodic paths parallelly on $(2td(\phi)+5)^n$ (polynomially many) different paths.

Now, we present how to solve $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KFS})$ using Lemma 7. From Lemma 4, we know that the number of minimal path schemas in a flat Kripke structure $S = \langle Q, \Delta, \mathbf{l} \rangle$ is finite and the length of a minimal path schema is at most $2 \times \mathrm{card}(\Delta)$. Hence, for solving the model-checking problem for a state q and a $\mathrm{PLTL}[\emptyset]$ formula ϕ, a possible algorithm consists in choosing non-deterministically a minimal path schema P starting at q and then apply the algorithm used to establish Lemma 7. This new algorithm would be in NP. Furthermore, thanks to Corollary 5, we know that if there exists a run ρ of S such that $\rho, 0 \models \phi$ then there exists a minimal path schema P such that ρ respects P. Consequently there is an algorithm in NP to solve $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KFS})$.

Theorem 10. $\mathrm{MC}(\mathrm{PLTL}[\emptyset], \mathcal{KFS})$ *is* NP*-complete.*

NP-hardness can be established as a variant of the proof of Lemma 8.

Similarly, $\mathcal{CPS}(k)$ denotes the class of path schemas obtained from flat counter systems with number of loops bounded by k.

Lemma 11. *For* $k \geq 2$, $\mathrm{MC}(\mathrm{PLTL}[\mathtt{C}], \mathcal{CPS}(k))$ *is* NP*-hard.*

The proof by reduction from SAT and it is less straightforward than the proof for Lemma 8 or the reduction presented in [16] when path schemas are involved. Indeed, we cannot encode the nondeterminism in the structure itself and the structure has only a constant number of loops. Actually, we cannot use a separate loop for each counter; the reduction is done by encoding the nondeterminism in the (possibly exponential) number of times a single loop is taken, and then using its binary encoding as an assignment for the propositional variables. Hence, the reduction uses in an essential way the counter values and the arithmetical constraints in the formula. By contrast, $\mathrm{MC}(\mathrm{PLTL}[\mathtt{C}], \mathcal{CPS}(1))$ can be shown in PTime.

5 Model-checking PLTL[$\mathtt{C}$] over Flat Counter Systems

In this section, we provide a nondeterministic polynomial-time algorithm to solve MC(PLTL[$\mathtt{C}$], $\mathcal{CFS}$) (see Algorithm 1). To do so, we combine Theorem 3 with small solutions of constraint systems.

5.1 Characterizing Runs by System of Equations

In this section, we show how to build a system of equations from a path schema P and a configuration c_0 such that the system of equations encodes the set of all runs respecting P from c_0. This can be done for path schemas without disjunctions in guards that satisfy an additional *validity* property. A path schema $P = p_1 l_1^+ p_2 l_2^+ \dots p_k l_k^\omega$ is *valid* whenever $\mathit{effect}(l_k)[i] \geq 0$ for every $i \in [1, n]$ (see Section 4 for the definition of $\mathit{effect}(l_k)$) and if all the guards in transitions in l_k are conjunctions of atomic guards, then for each guard occurring in the loop l_k of the form $\sum_i a_i \mathtt{x}_i \sim b$ with $\sim \in \{\leq, <\}$ [resp. with $\sim \in \{=\}$, with $\sim \in \{\geq, >\}$] , we have $\sum_i a_i \times \mathit{effect}(l_k)[i] \leq 0$ [resp. $\sum_i a_i \times \mathit{effect}(l_k)[i] = 0$, $\sum_i a_i \times \mathit{effect}(l_k)[i] \geq 0$]. It is easy to check that these conditions are necessary to visit the last loop l_k infinitely. More specifically, if a path schema is not valid, then no infinite run can respect it. Moreover, given a path schema, one can decide in polynomial time whether it is valid.

Now, let us consider a (not necessarily minimal) valid path schema $P = p_1 l_1^+ p_2 l_2^+ \dots p_k l_k^\omega$ ($k \geq 1$) obtained from a flat counter system S such that all the guards on transitions are conjunctions of atomic guards of the form $\sum_i a_i \mathtt{x}_i \sim b$ where $a_i \in \mathbb{Z}$, $b \in \mathbb{Z}$ and $\sim \in \{=, \leq, \geq, <, >\}$. Hence, disjunctions are *disallowed* in guards. The goal of this section (see Lemma 12 below) is to characterize the set $\mathit{iter}_P(c_0) \subseteq \mathbb{N}^{k-1}$ for some configuration c_0 as the set of solutions of a constraint system. For each loop l_i, we introduce a variable $\mathtt{y}_i$, whence the number of variables of the system/formula is precisely $k - 1$. A *constraint system* $\mathcal{E}$ over the set of variables $\{\mathtt{y}_1, \dots, \mathtt{y}_n\}$ is a quantifier-free Presburger formula built over $\{\mathtt{y}_1, \dots, \mathtt{y}_n\}$ as a conjunction of atomic constraints of the form $\sum_i a_i \mathtt{y}_i \sim b$ where $a_i, b \in \mathbb{Z}$ and $\sim \in \{=, \leq, \geq, <, >\}$. Conjunctions of atomic counter constraints and constraint systems are essentially the same objects but the distinction allows to emphasize the different purposes: guard on counters in operational models and symbolic representation of sets of tuples.

Lemma 12. *Let $S = \langle Q, \mathtt{C}_n, \Delta, \mathbf{l} \rangle$ be a flat counter system without disjunctions in guards, P be a valid path schema and c_0 be a configuration. One can compute in polynomial time a constraint system $\mathcal{E}$ such that the set of solutions of $\mathcal{E}$ is equal to $\mathit{iter}_P(c_0)$, $\mathcal{E}$ has $\mathrm{nbloops}(P) - 1$ variables, $\mathcal{E}$ has at most $\mathrm{len}(P) \times 2 \times \mathrm{size}(S)^2$ conjuncts and the greatest absolute value from constants in $\mathcal{E}$ is bounded by $n \times \mathrm{nbloops}(P) \times K^4 \times \mathrm{len}(P)^3$ where K is the greatest absolute value for constants occurring in S.*

5.2 Elimination of Arithmetical Constraints and Disjunctions

As stated in Lemma 12, the procedure for characterizing infinite runs in a counter system by a system of equations works only for a flat counter system with no

disjunction in guards (convexity of guards is essential). In this section, we show how to obtain such a system from a general flat counter system. Given a flat counter system $S = \langle Q, \mathtt{C}_n, \Delta, \mathbf{l}\rangle$, a configuration $c_0 = \langle q_0, \boldsymbol{v_0}\rangle$ and a minimal path schema P starting from the configuration c_0, we show that it is possible to build a finite set Y_P of path schemas such that (1) each path schema in Y_P has transitions without disjunctions in guards, (2) existence of a run ρ respecting P is equivalent to the existence of a path schema in Y_P having a run similar to ρ respecting it and (3) each path schema in Y_P is obtained from P by unfolding loops so that the terms in each loop satisfy the same atomic guards. Note that disjunctions could be easily eliminated at the cost of adding new transitions between states but this type of transformation may easily destroy flatness. Hence, the necessity to present a more sophisticated elimination procedure

We first introduce a few definitions. A (syntactic) *resource* R is a triple $\langle X, T, B\rangle$ such that X is a finite set of propositional variables, T is a finite set of terms $\mathtt{t}$ appearing in some guards of the form $\mathtt{t} \sim b$ (with $b \in \mathbb{Z}$) and B is a finite set of integers. We say that a resource $\mathsf{R} = \langle X, T, B\rangle$ is *coherent* with a counter system S [resp. with a path schema P] if B contains all the constants b occurring in guards of S [resp. of P] of the form $\mathtt{t} \sim b$ and T contains all the terms $\mathtt{t}$ occurring in guards of S [resp. of P] of the form $\mathtt{t} \sim b$. The resource R is coherent with a formula $\phi \in \mathrm{PLTL}[\mathtt{C}]$, whenever the atomic formulae of ϕ are either of the form $p \in X$ or $\mathtt{t} \sim b$ with $\mathtt{t} \in T$ and $b \in B$. In the sequel, we assume that the considered resource is always coherent with S.

Assuming that $B = \{b_1, \ldots, b_m\}$ with $b_1 < \cdots < b_m$, we write I to denote the finite set of intervals $I = \{(-\infty, b_1 - 1], [b_1, b_1], [b_1 + 1, b_2 - 1], [b_2, b_2], \cdots, [b_m, b_m], [b_m + 1, \infty)\}$. Note that I contains exactly $2m+1$ intervals. A *term map* $\boldsymbol{m}$ is a map $\boldsymbol{m} : T \to I$ that abstracts term values. A *footprint* is an abstraction of a model for $\mathrm{PLTL}[\mathtt{C}]$ restricted to elements from the resource R: it is of the form $\mathsf{ft} : \mathbb{N} \to 2^X \times I^T$ where I is the set of intervals built from B. The satisfaction relation $\models$ involving models or runs can be adapted to footprints as follows (formulae and footprints are from the same resource):

- $\mathsf{ft}, i \models_{\mathsf{symb}} p \stackrel{\text{def}}{\Leftrightarrow} p \in \pi_1(\mathsf{ft}(i))$; $\mathsf{ft}, i \models_{\mathsf{symb}} \mathtt{t} \geq b \stackrel{\text{def}}{\Leftrightarrow} \pi_2(\mathsf{ft}(i))(\mathtt{t}) \subseteq [b, +\infty)$,
- $\mathsf{ft}, i \models_{\mathsf{symb}} \mathtt{t} \leq b \stackrel{\text{def}}{\Leftrightarrow} \pi_2(\mathsf{ft}(i))(\mathtt{t}) \subseteq (-\infty, +b]$,
- $\mathsf{ft}, i \models_{\mathsf{symb}} \mathtt{X}\phi \stackrel{\text{def}}{\Leftrightarrow} \mathsf{ft}, i+1 \models_{\mathsf{symb}} \phi$,
- $\mathsf{ft}, i \models_{\mathsf{symb}} \phi \mathtt{U} \psi \stackrel{\text{def}}{\Leftrightarrow} \exists j \geq i$ s.t. $\mathsf{ft}, j \models_{\mathsf{symb}} \psi$ and $\forall j' \in [i, j-1]$, $\mathsf{ft}, j' \models_{\mathsf{symb}} \phi$.

We omit the other obvious clauses. $\models_{\mathsf{symb}}$ is the satisfaction relation for Past LTL when arithmetical constraints are understood as abstract propositions. Let $\mathsf{R} = \langle X, T, B\rangle$ be a resource and $\rho = \langle q_0, \boldsymbol{v_0}\rangle, \langle q_1, \boldsymbol{v_1}\rangle \cdots$ be an infinite run of S. The *footprint* of ρ with respect to R is the footprint $\mathsf{ft}(\rho)$ such that for $i \geq 0$, we have $\mathsf{ft}(\rho)(i) \stackrel{\text{def}}{=} \langle \mathbf{l}(q_i) \cap X, \boldsymbol{m}_i\rangle$ where for every term $\mathtt{t} = \sum_j a_j \mathtt{x}_j \in T$, we have $\sum_j a_j \boldsymbol{v_i}[j] \in \boldsymbol{m}_i(\mathtt{t})$. Note that $\sum_j a_j \boldsymbol{v_i}[j]$ belongs to a unique element of I since I is a partition of $\mathbb{Z}$. Hence, this definition makes sense. Lemma 13 below roughly states that satisfaction of a formula on a run can be checked symbolically from the footprint (this is useful for the correctness of forthcoming Algorithm 1).

Lemma 13. *Let ϕ be in* PLTL[C], $\mathsf{R} = \langle X, T, B\rangle$ *be coherent with* ϕ, $\rho = \langle q_0, \boldsymbol{v_0}\rangle, \langle q_1, \boldsymbol{v_1}\rangle \cdots$ *be an infinite run and* $i \geq 0$. *(I) Then* $\rho, i \models \phi$ *iff* $\mathsf{ft}(\rho), i \models_{\mathsf{symb}} \phi$. *(II) If* ρ' *is an infinite run s.t.* $\mathsf{ft}(\rho) = \mathsf{ft}(\rho')$, *then* $\rho, i \models \phi$ *iff* $\rho', i \models \phi$.

In [6] we explain in details how to build a set Y_P of path schemas without disjunctions from a minimal path schema P, an initial configuration $\langle q_0, \boldsymbol{v_0}\rangle$ and a resource R. The main idea of this construction consists in adding to the control states of path schemas some information on the intervals to which belongs each term of T. In fact, in the transitions appearing in path schemas of Y_P the states belong to $Q' = Q \times I^T$. Before stating the properties of Y_P, we introduce some notations. Given $\mathtt{t} = \sum_j a_j \mathsf{x}_j \in T$, $\mathbf{u} \in \mathbb{Z}^n$ and a term map $\boldsymbol{m}$, we write $\psi(\mathtt{t}, \mathbf{u}, \boldsymbol{m}(\mathtt{t}))$ to denote the formula below ($b, b' \in B$): $\psi(\mathtt{t}, \mathbf{u}, (-\infty, b]) \stackrel{\text{def}}{=} \sum_j a_j(\mathsf{x}_j + \mathbf{u}(j)) \leq b$; $\psi(\mathtt{t}, \mathbf{u}, [b, +\infty)) \stackrel{\text{def}}{=} \sum_j a_j(\mathsf{x}_j + \mathbf{u}(j)) \geq b$ and $\psi(\mathtt{t}, \mathbf{u}, [b, b']) = ((\sum_j a_j(\mathsf{x}_j + \mathbf{u}(j)) \leq b') \wedge ((\sum_j a_j(\mathsf{x}_j + \mathbf{u}(j)) \geq b)$. We write $\mathtt{G}^\star(T, B, U)$ to denote the set of guards of the form $\psi(\mathtt{t}, \mathbf{u}, \boldsymbol{m}(\mathtt{t}))$ where $\mathtt{t} \in T$, U is the finite set of updates from P and $\boldsymbol{m} : T \to I$. Each guard in $\mathtt{G}^\star(T, B, U)$ is of linear size in the size of P. We denote $\tilde{\Delta}$ the set of transitions $Q' \times \mathtt{G}^\star(T, B, U) \times U \times Q'$. Note that the transitions in $\tilde{\Delta}$ do not contain guards with disjunctions and $\tilde{\Delta}$ is finite. We also define a function proj which associates to $w \in \tilde{\Delta}^\omega$ the ω-sequence $\mathsf{proj}(w) : \mathbb{N} \to 2^X \times I^T$ such that for all $i \in \mathbb{N}$, if $w(i) = \langle\langle q, \boldsymbol{m}\rangle, \mathsf{g}, \mathbf{u}, \langle q', \boldsymbol{m'}\rangle\rangle$ and $\mathbf{l}(q) \cap X = L$ then $\mathsf{proj}(w)(i) \stackrel{\text{def}}{=} \langle L, \boldsymbol{m}\rangle$.

We show that it is possible to build a finite set Y_P of path schemas over $\tilde{\Delta}$ such that if $P' = p'_1(l'_1)^+ p'_2(l'_2)^+ \ldots p'_{k'}(l'_{k'})^\omega$ is a path schema in Y_P and ρ is a run $\langle\langle q_0, \boldsymbol{m}_0\rangle, \boldsymbol{v_0}\rangle \to \langle\langle q_1, \boldsymbol{m}_1\rangle, \boldsymbol{v_1}\rangle \to \langle\langle q_2, \boldsymbol{m}_2\rangle, \boldsymbol{v_2}\rangle \cdots$ respecting P' we have that $\mathsf{proj}(lab(\rho)) = \mathsf{ft}(\rho)$. This point will be useful for Algorithm 1. The following theorem lists the main properties of the set Y_P.

Theorem 14. *Given a flat counter system S, a minimal path schema P, a resource $\mathsf{R} = \langle X, T, B\rangle$ coherent with P and a configuration $\langle q_0, \boldsymbol{v_0}\rangle$, there is a finite set of path schemas Y_P over $\tilde{\Delta}$ satisfying (1)–(6) below.*

1. *No path schema in Y_P contains guards with disjunctions in it.*
2. *There exists a polynomial $q^\star(\cdot)$ such that for every $P' \in Y_P$, $\mathrm{len}(P') \leq q^\star(\mathrm{len}(P) + \mathrm{card}(T) + \mathrm{card}(B))$.*
3. *Checking whether a path schema P' over $\tilde{\Delta}$ belongs to Y_P can be done in polynomial time in $\mathrm{size}(P) + \mathrm{card}(T) + \mathrm{card}(B)$.*
4. *For every run ρ respecting P and starting at $\langle q_0, \boldsymbol{v_0}\rangle$, we can find a run ρ' respecting some $P' \in Y_P$ such that $\rho \models \phi$ iff $\rho' \models \phi$ for every ϕ built over R.*
5. *For every run ρ' respecting some $P' \in Y_P$ with initial values $\boldsymbol{v_0}$, we can find a run ρ respecting P such that $\rho \models \phi$ iff $\rho' \models \phi$ for every ϕ built over R.*
6. *For every ultimately periodic word $w \cdot u^\omega \in \mathcal{L}(P')$, for every ϕ built over R checking whether $\mathsf{proj}(w \cdot u^\omega), 0 \models_{\mathsf{symb}} \phi$ can be done in polynomial time in the size of $w \cdot u$ and in the size of ϕ.*

5.3 Main Algorithm

In Algorithm 1 below, a polynomial $p^\star(\cdot)$ is used. We can define polynomial $p^\star(\cdot)$ using the small solutions for constraint systems [2], see details in [6]. Note that

$\boldsymbol{y'}$ is a refinement of $\boldsymbol{y}$ (for all i, we have $\boldsymbol{y'}[i] \approx_{2td(\phi)+5} \boldsymbol{y}[i]$) in which counter values are taken into account.

Algorithm 1 The main algorithm in NP with inputs S, $c_0 = \langle q, \boldsymbol{v_0} \rangle$, ϕ

1: guess a minimal path schema P of S
2: build a resource $\mathsf{R} = \langle X, T, B \rangle$ coherent with P and ϕ
3: guess a valid schema $P' = p_1 l_1^+ p_2 l_2^+ \dots p_k l_k^\omega$ such that $\text{len}(P') \leq q^\star(\text{len}(P) + \text{card}(T) + \text{card}(B))$
4: guess $\boldsymbol{y} \in [1, 2td(\phi)+5]^{k-1}$; guess $\boldsymbol{y'} \in [1, 2^{p^\star(\text{size}(S)+\text{size}(c_0)+\text{size}(\phi))}]^{k-1}$
5: check that P' belongs to Y_P
6: check that $\mathsf{proj}(p_1 l_1^{\boldsymbol{y}[1]} p_2 l_2^{\boldsymbol{y}[2]} \dots l_{k-1}^{\boldsymbol{y}[k-1]} p_k l_k^\omega), 0 \models_{\mathsf{symb}} \phi$
7: build $\mathcal{E}$ over $\mathsf{y}_1, \dots, \mathsf{y}_{k-1}$ for P' with initial values $\boldsymbol{v_0}$ (obtained from Lemma 12)
8: **for** $i = 1 \to k-1$ **do**
9: **if** $\boldsymbol{y}[i] = 2td(\phi)+5$ **then** $\psi_i \leftarrow$ "$\mathsf{y}_i \geq 2td(\phi)+5$" **else** $\psi_i \leftarrow$ "$\mathsf{y}_i = \boldsymbol{y}[i]$"
10: **end for**
11: check that $\boldsymbol{y'} \models \mathcal{E} \wedge \psi_1 \wedge \dots \wedge \psi_{k-1}$

Algorithm 1 starts by guessing a path schema P (line 1) and an unfolded path schema $P' = p_1 l_1^+ p_2 l_2^+ \dots p_k l_k^\omega$ (line 3) and check whether P' belongs to Y_P (line 5). It remains to check whether there is a run ρ respecting P' such that $\rho \models \phi$. Suppose there is such a run ρ; let $\boldsymbol{y}$ be the unique tuple in $[1, 2td(\phi)+5]^{k-1}$ such that $\boldsymbol{y} \approx_{2td(\phi)+5} iter_{P'}(\rho)$. By Proposition 6, we have $\mathsf{proj}(p_1 l_1^{\boldsymbol{y}[1]} p_2 l_2^{\boldsymbol{y}[2]} \dots l_{k-1}^{\boldsymbol{y}[k-1]} p_k l_k^\omega), 0 \models_{\mathsf{symb}} \phi$. Since the set of tuples of the form $iter_{P'}(\rho)$ is characterized by a system of equations, by the existence of small solutions from [2], we can assume that $iter_{P'}(\rho)$ contains only small values. Hence line 4 guesses $\boldsymbol{y}$ and $\boldsymbol{y'}$ (corresponding to $iter_{P'}(\rho)$ with small values). Line 6 precisely checks $\mathsf{proj}(p_1 l_1^{\boldsymbol{y}[1]} p_2 l_2^{\boldsymbol{y}[2]} \dots l_{k-1}^{\boldsymbol{y}[k-1]} p_k l_k^\omega), 0 \models_{\mathsf{symb}} \phi$ whereas line 11 checks whether $\boldsymbol{y'}$ encodes a run respecting P' with $\boldsymbol{y'} \approx_{2td(\phi)+5} \boldsymbol{y}$.

Lemma 15. *Algorithm 1 runs in nondeterministic polynomial time.*

It remains to check that Algorithm 1 is correct.

Lemma 16. *$S, c_0 \models \phi$ iff Algorithm 1 on inputs S, c_0, ϕ has an accepting run.*

In the proof of Lemma 16, we take advantage of all our preliminary results.

Proof. By way of example, we show that if Algorithm 1 on inputs S, $c_0 = \langle q_0, \boldsymbol{v_0} \rangle$, ϕ has an accepting computation, then $S, c_0 \models \phi$. This means that there are P, P', $\boldsymbol{y}$, $\boldsymbol{y'}$ that satisfy all the checks. Let $w = p_1 l_1^{\boldsymbol{y'}[1]} \cdots p_{k-1} l_{k-1}^{\boldsymbol{y'}[k-1]} p_k l_k^\omega$ and $\rho = \langle\langle q_0, \boldsymbol{m_0} \rangle, \boldsymbol{v_0}\rangle\langle\langle q_1, \boldsymbol{m_1}\rangle, \boldsymbol{x_1}\rangle\langle\langle q_2, \boldsymbol{m_2}\rangle, \boldsymbol{x_2}\rangle \cdots \in (Q' \times \mathbb{Z}^n)^\omega$ be defined as follows: for every $i \geq 0$, $q_i \stackrel{\text{def}}{=} \pi_1(source(w(i)))$, and for every $i \geq 1$, we have $\boldsymbol{x_i} \stackrel{\text{def}}{=} \boldsymbol{x_{i-1}} + update(w(i))$. By Lemma 12, since $\boldsymbol{y'} \models \mathcal{E} \wedge \psi_1 \wedge \dots \wedge \psi_{k-1}$, ρ is a run respecting P' starting at the configuration $\langle\langle q_0, \boldsymbol{m_0}\rangle, \boldsymbol{v_0}\rangle$. Since $\boldsymbol{y'} \models \psi_1 \wedge \dots \wedge \psi_{k-1}$ and $\boldsymbol{y} \models \psi_1 \wedge \dots \wedge \psi_{k-1}$, by Proposition 6, (✠) $\mathsf{proj}(p_1 l_1^{\boldsymbol{y}[1]} p_2 l_2^{\boldsymbol{y}[2]} \dots l_{k-1}^{\boldsymbol{y}[k-1]} p_k l_k^\omega), 0 \models_{\mathsf{symb}} \phi$, iff (✠✠) $\mathsf{proj}(p_1 l_1^{\boldsymbol{y'}[1]} p_2 l_2^{\boldsymbol{y'}[2]} \dots l_{k-1}^{\boldsymbol{y'}[k-1]} p_k l_k^\omega), 0 \models_{\mathsf{symb}} \phi$. Algorithm 1 guarantees that $\mathsf{proj}(p_1 l_1^{\boldsymbol{y}[1]} p_2 l_2^{\boldsymbol{y}[2]} \dots l_{k-1}^{\boldsymbol{y}[k-1]} p_k l_k^\omega), 0 \models_{\mathsf{symb}} \phi$, whence we have (✠✠). Since $\mathsf{proj}(p_1 l_1^{\boldsymbol{y'}[1]} p_2 l_2^{\boldsymbol{y'}[2]} \dots l_{k-1}^{\boldsymbol{y'}[k-1]} p_k l_k^\omega) = \mathsf{ft}(\rho)$, by Lemma 13, we deduce that $\rho, 0 \models$

ϕ. By Theorem 14(5), there is an infinite run ρ', starting at the configuration $\langle q_0, \boldsymbol{v_0} \rangle$ and respecting P, such that $\rho', 0 \models \phi$.

For the other direction, see [6]. □

As a corollary, we can state the main result of the paper.

Theorem 17. MC(PLTL[C], $\mathcal{CFS}$) *is* NP*-complete.*

6 Conclusion

Classes of Systems	PLTL[$\emptyset$]	PLTL[C]	Reachability
$\mathcal{KPS}$	NP-complete See [16] for X and U	—	PTime
$\mathcal{CPS}$	NP-complete	NP-complete (Theo. 17)	NP-complete
$\mathcal{KPS}(n)$	PTime (Theo. 9)	—	PTime
$\mathcal{CPS}(n)$, $n > 1$	??	NP-complete (Lem. 11)	??
$\mathcal{CPS}(1)$	PTime	PTime	PTime
$\mathcal{KFS}$	NP-complete See [16] for X and U	—	PTime
$\mathcal{CFS}$	NP-complete	**NP-complete** (Theo. 17)	NP-complete

We have investigated the computational complexity of the model-checking problem for flat counter systems with formulae from an enriched version of LTL. Our main result is the NP-completeness of MC(PLTL[C], $\mathcal{CFS}$), significantly improving the complexity upper bound from [7]. This also improves the results about the effective semilinearity of the reachability relations for such flat counter systems from [5,11] and it extends the recent result on the NP-completeness of model-checking flat Kripke structures with LTL from [16] by adding counters and past-time operators. Our main results are presented above and compared to the reachability problem.

As far as the proof technique is concerned, the NP upper bound is obtained as a combination of a general stuttering property for LTL with past-time operators (a result extending what is done in [17] with past-time operators) and the use of small integer solutions for quantifier-free Presburger formulae [2]. There are several related problems which are not addressed in the paper. For instance, the extension of the model-checking problem to full CTL* is known to be decidable [7] but the characterization of its exact complexity is open.

References

1. B. Boigelot. *Symbolic methods for exploring infinite state spaces.* PhD thesis, Université de Liège, 1998.
2. I. Borosh and L. Treybig. Bounds on positive integral solutions of linear Diophantine equations. *American Mathematical Society*, 55:299–304, 1976.
3. M. Bozga, R. Iosif, and F. Konecný. Fast acceleration of ultimately periodic relations. In *CAV'10*, volume 6174 of *LNCS*, pages 227–242. Springer, 2009.

4. H. Comon and V. Cortier. Flatness is not a weakness. In *CSL'00*, volume 1862 of *LNCS*, pages 262–276. Springer, 2000.
5. H. Comon and Y. Jurski. Multiple counter automata, safety analysis and PA. In *CAV'98*, volume 1427 of *LNCS*, pages 268–279. Springer, 1998.
6. S. Demri, A. Dhar, and A. Sangnier. Taming past LTL and flat counter systems. Technical report, 2012. ArXiv.
7. S. Demri, A. Finkel, V. Goranko, and G. van Drimmelen. Model-checking CTL* over flat Presburger counter systems. *JANCL*, 20(4):313–344, 2010.
8. C. Dixon, M. Fisher, and B. Konev. Temporal logic with capacity constraints. In *FROCOS'07*, volume 4720 of *LNCS*, pages 163–177. Springer, 2007.
9. J. Esparza, A. Finkel, and R. Mayr. On the verification of broadcast protocols. In *LICS'99*, pages 352–359, 1999.
10. K. Etessami and T. Wilke. An until hierarchy and other applications of an Ehrenfeucht-Fraïssé game for temporal logic. *I&C*, 160(1–2):88–108, 2000.
11. A. Finkel and J. Leroux. How to compose Presburger accelerations: Applications to broadcast protocols. In *FST&TCS'02*, volume 2256 of *LNCS*, pages 145–156. Springer, 2002.
12. A. Finkel, E. Lozes, and A. Sangnier. Towards model-checking programs with lists. In *Infinity in Logic & Computation*, volume 5489 of *LNAI*, pages 56–82. Springer, 2009.
13. D. Gabbay. The declarative past and imperative future. In *Temporal Logic in Specification, Altrincham, UK*, volume 398 of *LNCS*, pages 409–448. Springer, 1987.
14. C. Haase, S. Kreutzer, J. Ouaknine, and J. Worrell. Reachability in succinct and parametric one-counter automata. In *CONCUR'09*, volume 5710 of *LNCS*, pages 369–383. Springer, 2009.
15. L. Kuhtz. *Model Checking Finite Paths and Trees.* PhD thesis, Universität des Saarlandes, 2010.
16. L. Kuhtz and B. Finkbeiner. Weak Kripke structures and LTL. In *CONCUR'11*, volume 6901 of *LNCS*, pages 419–433. Springer, 2011.
17. A. Kučera and J. Strejček. The stuttering principle revisited. *Acta Informatica*, 41(7–8):415–434, 2005.
18. F. Laroussinie, N. Markey, and P. Schnoebelen. Temporal logic with forgettable past. In *LICS'02*, pages 383–392. IEEE, 2002.
19. F. Laroussinie and P. Schnoebelen. Specification in CTL + past for verification in CTL. *I&C*, 156:236–263, 2000.
20. J. Leroux and G. Sutre. Flat counter systems are everywhere! In *ATVA'05*, volume 3707 of *LNCS*, pages 489–503. Springer, 2005.
21. C. Lutz. NEXPTIME-complete description logics with concrete domains. In *IJCAR'01*, volume 2083 of *LNCS*, pages 46–60. Springer, 2001.
22. M. Minsky. *Computation, Finite and Infinite Machines.* Prentice Hall, 1967.
23. D. Peled and T. Wilke. Stutter-invariant temporal properties are expressible without the next-time operator. *IPL*, 63:243–246, 1997.
24. C. Rackoff. The covering and boundedness problems for vector addition systems. *TCS*, 6(2):223–231, 1978.